SURFACE MESH PROJECTION FOR HEXAHEDRAL MESH GENERATION BY SWEEPING

Xevi Roca[1] Josep Sarrate[1] Antonio Huerta[1]

[1]*Laboratori de Càlcul Numèric (LaCàN), Departament de Matemàtica Aplicada III, Universitat Politècnica de Catalunya, Barcelona, Spain. xevi.roca@upc.es*

ABSTRACT

Sweep method is one of the most robust techniques to generate hexahedral meshes in extrusion volumes. One of the main issues to be dealt by any sweep algorithm is the projection of a source surface mesh onto the target surface. This paper presents a new algorithm to map a given mesh over the source surface onto the target surface. This projection is carried out by means of a least-squares approximation of an affine mapping defined between the parametric spaces of the surfaces. Once the new mesh is obtained on the parametric space of the target surface, it is mapped up according to the target surface parameterization. Therefore, the developed algorithm does not require to solve any root finding problem to ensure that the projected nodes are on the target surface. Afterwards, this projection algorithm is extended to three dimensional cases and it is used to generate the inner layers of elements in the physical space.

Keywords: Finite element method, mesh generation, sweep, hexahedral elements.

1. INTRODUCTION

Hexahedral mesh generation has been an important and active research area over the last decade. Several algorithms has been devised in order to generate hexahedral meshes for any arbitrary geometry such as (see [1, 2] for a detailed survey): grid based methods [3, 4, 5], decomposition based approaches [6, 7, 8], advancing front techniques [9], or dual methods [10, 11]. However, a fully automatic hexahedral mesh generation algorithm for any arbitrary geometry is still way off. Moreover, further research has still to be developed in order to work out a general purpose algorithm that, given any volume, generates high quality hexahedral elements at low cost (both in cpu and in user interaction time).

Therefore, special attention has been focused on existing algorithms that decompose the entire geometry into several simpler pieces (assembly models). These smaller volumes can be easily meshed by well-known methods that obtain an outstanding performance in these simpler volumes, such as: mapping [12], submapping [13], and sweeping [14, 15, 16, 17, 18].

Most of the CAD commercial packages allow generation of volumes by extrude, or sweep, a surface along a delimited axis. These one-to-one sweep volumes are defined by a *source surface*, a *target surface* and a series of *linking-sides* (see figure 1). Note that the source and target surfaces may have different areas and curvatures. Moreover, there are no restrictions on the number of edges that describe these surfaces. However, they must be topologically equivalent. That is, they must have the same number of holes and logical sides. Furthermore, linking-sides have to be mappable. Hence, having only four logical sides.

Taking into account the definition of an extrusion geometry, it is reasonable to generate the volume mesh by sweeping a given discretization of the source surface along the volume until it reaches the target surface. Therefore, the traditional procedure to generate an all–hexahedral mesh by sweeping is decomposed in the following four steps:

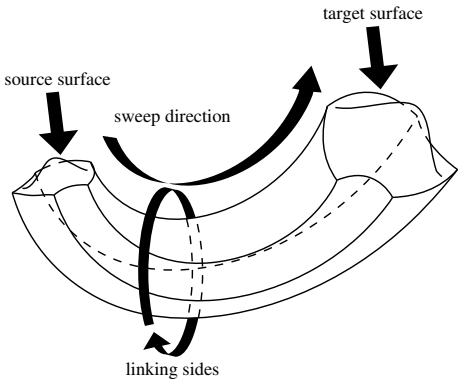

Figure 1: Example of a two and a half dimensional volume.

1. Generation of a quadrilateral mesh over the source surface.
2. Projection of the source mesh onto the target surface (ensuring that both source and target mesh have same connectivity).
3. Generation of a structured quadrilateral mesh over the linking-sides.
4. Generation of the inner layers of nodes and elements.

Note that surface mesh generation is involved in the first three steps. However, hexahedral elements are only generated in the last one. Several robust quadrilateral surface mesh generation algorithms have been developed which greatly simplify the meshing process involved in the first step [19, 20, 21, 22]. The gridding of the linking-sides involved in the third step can be generated using any standard structured quadrilateral surface mesh generator [2, 23]. Hence, the two main issues to be dealt by any sweep algorithm are the second and fourth step. In both steps, the meshes to be generated (i.e. the mesh over the target surface and the inner layer meshes) must have the same connectivity as the source surface mesh.

Several algorithms have been developed to map meshes between surfaces [24]. Most of them involve an orthogonal projection of nodes onto the target surface. Note that these projections are expensive from a computational point of view since it is necessary to solve as many root finding problems as internal points are on the grid of the source surface. In order to overcome this drawback, this paper presents a new and efficient algorithm to map a given mesh over the source surface onto the target surface. This projection is determined by means of a least-squares approximation of an affine mapping defined between the parametric representation of the loops of boundary nodes of the

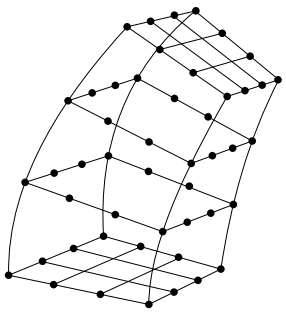

Figure 2: Available data for generating the inner layers of nodes.

cap surfaces. Once the new mesh is obtained on the parametric space of the target surface, it is mapped up according to the target surface parameterization.

The developed algorithm to map meshes between cap surfaces can not be directly applied in order to generate the inner layer of nodes, since these layers are not defined by parametric surfaces. In fact, the available data to determine the position of the inner layers of nodes is: *1.*- the loops of nodes on the linking surfaces, and *2.*- the cap surface meshes (see figure 2). Hence, the previous projection algorithm is extended to three dimensional space and it is used to generate the inner layers of elements in the physical space. The obtained algorithm becomes analogous to the method proposed in [14]. In order to improve the quality of the meshes generated on each layer, the inner nodes are located using a weighted least-squares approximation of the transformation between the boundary nodes of the cap surfaces and the boundary nodes of the layer as in [16].

2. SWEEP ALGORITHM

In order to ensure that the one-to-one sweep algorithm can be applied, the following requirements have to be accomplished:

1. Source and target surfaces must be topologically equivalent (they must have the same number of holes and logical sides).
2. Linking-sides must be mappable, or equivalently, defined by four logical sides.
3. Sweep volume has one source surface and one target surface.
4. Sweep volume must be defined by only one axis.

A detailed presentation on constraints which must be met for a volume to be sweepable, in a generic sense, are presented in [25].

Notice that the algorithm could be generalized to multi-source / multi-target geometries following the cooper tool algorithm [16, 17]. Moreover, it may also be extended to multi-axis sweep directions according to [26]. However, these objectives are beyond the scope of the present work.

Sweep volumes are usually defined by commercial CAD applications. Hence, source and target surfaces may be trimmed surfaces (they allow a more flexible description of surfaces with holes or complex boundaries). In order to properly deduce the projection algorithm between cap surfaces, some details of the definition of trimmed surfaces have to be revised. In particular, it is important to point out that the domain of a trimmed surface, in general, is not a rectangle $[a,b]\times[c,d]\subset\mathbb{R}^2$ in the parametric space. To be more accurate, let $\varphi: V\subset\mathbb{R}^2\to\mathbb{R}^3$ be the parametric definition of a surface, where V is an open and bounded set of the plane. Let $D_S\subset V$ be a closed subset of V. Then, a trimmed surface, S, is the image of the restriction of φ at the subset D_S. That is, $\varphi|_{D_S}: D_S\subset V\to S\subset\mathbb{R}^3$, where $S=\varphi(D_S)$.

From here to the end of this work we assume that a geometrical kernel is available, such that surfaces are parameterized. In fact, this kernel must provide query functions for obtaining the physical coordinates of a parametric point. This requirement is necessary if the parametric space projection algorithm, presented in section 2.2, has to be implemented.

2.1 Source surface mesh generation

In standard industrial applications, surfaces are defined using any CAD software. Therefore, the quadrilateral mesh generator used to discretize the source surface has to be able to work with trimmed surfaces. In this paper, the source surface is discretized using an extended version to parametric surfaces of a previously developed unstructured quadrilateral mesh generator [21, 22].

2.2 Target surface mesh generation: the projection algorithm

Once the source surface, S, is meshed the next step is to map it onto the target surface, T. As it has been previously noted, most of the developed algorithms to map meshes between surfaces have to solve several root finding problems. In order to overcome this drawback a new and efficient algorithm is devised to map meshes between trimmed surfaces. In fact, the mapping is defined between the parametric spaces of the surfaces, D_S and D_T. It will be proved that determining this mapping is equivalent to finding a projection of meshes between surfaces in the physical domain. Then, the obtained mesh is mapped up according to the target surface parameterization. For some surfaces, it may be necessary to smooth the new surface mesh. Note that this smoothing is also needed in other methods [24].

First of all, we will show that determining a projection between trimmed surfaces is equivalent to finding out a projection between their parametric spaces. To this end, assume that the source and target surfaces are trimmed surfaces. Let

$$\begin{aligned}\varphi_S:&\quad V_S\subset\mathbb{R}^2\to\mathbb{R}^3\\ \varphi_T:&\quad V_T\subset\mathbb{R}^2\to\mathbb{R}^3\end{aligned}$$

be their extended parameterization, where V_S and V_T are two open and bounded sets of $\mathbb{R}^2$. If φ_S and φ_T are continuous and injective, the Brouwer's theorem on invariance of domain [27] states that they are also open mappings. Since they are open mappings, their restrictions (that is, the definition of the trimmed surfaces)

$$\begin{aligned}\varphi_S|_{D_S}:&\quad D_S\subset V_S\to S\subset\mathbb{R}^3,\qquad S:=\varphi_S(D_S)\\ \varphi_T|_{D_T}:&\quad D_T\subset V_T\to T\subset\mathbb{R}^3,\qquad T:=\varphi_T(D_T)\end{aligned}$$

are homeomorphisms in D_S and D_T respectively.

Recall that our aim is to determine a mapping $\widetilde{\psi}: S\to T$ such that, given a mesh, $\mathcal{M}_S$, over the source surface, it yields a mesh, $\mathcal{M}_T$, onto the target surface with the same connectivities. Since S and T have the same topology it is reasonable to assume that $\widetilde{\psi}$ is also an homeomorphism.

Since $\varphi_S|_{D_S}$, $\varphi_T|_{D_T}$ and $\widetilde{\psi}$ are homeomorphisms, it is possible to define

$$\psi:=\varphi_T|_{D_T}^{-1}\circ\widetilde{\psi}\circ\varphi_S|_{D_S},$$

such that

$$\begin{array}{ccc} S\subset\mathbb{R}^3 & \xrightarrow{\ \widetilde{\psi}\ } & T\subset\mathbb{R}^3\\ \varphi_S|_{D_S}\uparrow & & \uparrow\varphi_T|_{D_T}\\ D_S\subset\mathbb{R}^2 & \xrightarrow{\ \psi\ } & D_T\subset\mathbb{R}^2\end{array}\qquad(1)$$

Under these conditions, the diagram of mappings (1) is a commutative diagram. Hence, it is feasible to find first the projection ψ, homeomorphism between the parametric domains D_S and D_T, and then, mapping up the new mesh onto the target surface, T, according to its parameterization, $\varphi_T|_{D_T}$, as

$$\widetilde{\psi}=\varphi_T|_{D_T}\circ\psi\circ{\varphi_S|_{D_S}}^{-1}.\qquad(2)$$

Note that it is not required to deduce the analytical expression of the inverse function ${\varphi_S|_{D_S}}^{-1}$. It suffices to store the pre-images of nodal coordinates of $\mathcal{M}_S$ by $\varphi_S|_{D_S}$. This can be achieved if the application stores both the physical and the parametric coordinates of each mesh node.

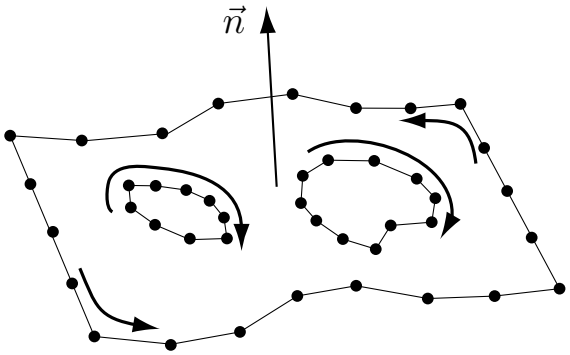

Figure 3: Boundary nodes of a non simple connected surface.

Therefore, special attention has to be focused on to settle the projection between D_S and D_T from the available data. Let n, with $n \geq 3$, be the number of nodes on all the boundary loops of the cap surfaces. We assume that each cap surface is delimited by one outer boundary and one inner boundary for each hole. These boundaries are previously meshed, and a series of loops of nodes on the boundary of the surface are obtained (see figure 3). Let $U_S = \{u_S^i\}_{i=1,\dots,n} \subset \mathbb{R}^2$ and $U_T = \{u_T^i\}_{i=1,\dots,n} \subset \mathbb{R}^2$ be the parametric coordinates of all boundary nodes of the source and target surfaces, respectively. It is important to point out that the physical coordinates of these points (i.e. their images by $\varphi_S|_{D_S}$ and $\varphi_T|_{D_T}$) do not necessarily determine planar loops. The goal is to find a function ψ such that

$$\psi(u_S^i) = u_T^i, \qquad i = 1, \dots, n. \tag{3}$$

Notice that it has only been required that the function ψ be a homeomorphism.

$\vec{n}$ In this algorithm, the homeomorphism ψ is approximated by an affine mapping

$$u_T \;=\; \psi(u_S) \;\approx\; \mathbf{A}(u_S - u_S^{arb}) + \mathbf{b}, \tag{4}$$

where u_S and u_T are points on D_S and D_T respectively, $\mathbf{A}$ is a linear transformation with the origin at one arbitrary point u_S^{arb} and $\mathbf{b}$ is a translation vector. Unfortunately, given any two loops of boundary data, there is not an affine mapping that verifies (3). Therefore, we look for a linear transformation, $\mathbf{A}$, and a translation vector $\mathbf{b}$ that fits in the least-squares sense the conditions (3). Hence, $\mathbf{A}$ and $\mathbf{b}$ are such that minimizes

$$F(\mathbf{A}, \mathbf{b}) = \sum_{i=1}^{n} \left\| u_T^i - \Big(\mathbf{A}(u_S^i - u_S^{arb}) + \mathbf{b} \Big) \right\|^2. \tag{5}$$

It is straightforward to show that if

$$u_S^{arb} \;:=\; u_S^c \;=\; \frac{1}{n} \sum_{i=1}^{n} u_S^i,$$

then

$$\mathbf{b} \;=\; u_T^c \;=\; \frac{1}{n} \sum_{i=1}^{n} u_T^i. \tag{6}$$

Therefore, the following coordinates are defined

$$\overline{u}_S = u_S - u_S^c, \qquad \overline{u}_T = u_T - u_T^c, \tag{7}$$

such that (4) can be written as

$$\overline{u}_T \;=\; \overline{\psi}(\overline{u}_S) \;\approx\; \mathbf{A}\overline{u}_S, \tag{8}$$

where $\overline{\psi}$ is the expression of ψ in the new coordinates. Now the minimization problem (5) is

$$F(\mathbf{A}) = \sum_{i=1}^{n} \left\| \overline{u}_T^i - \mathbf{A}\overline{u}_S^i \right\|^2. \tag{9}$$

Since $\mathbf{A}$ is a linear transformation, it can be written as

$$\mathbf{A} = \sum_{i=1}^{4} \lambda_i \mathbf{B}_i, \tag{10}$$

where

$$\begin{aligned} \mathbf{B}_1 &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \qquad \mathbf{B}_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \\ \mathbf{B}_3 &= \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \qquad \mathbf{B}_4 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \end{aligned} \tag{11}$$

is a base of the linear transformations from $\mathbb{R}^2$ to $\mathbb{R}^2$, and $\lambda_i \in \mathbb{R}$, with $i = 1, \dots, 4$.

If the scalar product of two functions

$$\begin{aligned} \boldsymbol{f} &: \; U_S \subset \mathbb{R}^2 \to \mathbb{R}^2 \\ \boldsymbol{g} &: \; U_S \subset \mathbb{R}^2 \to \mathbb{R}^2, \end{aligned}$$

is defined as

$$\begin{aligned} < \boldsymbol{f}, \boldsymbol{g} > \;&:=\; \sum_{i=1}^{n} < \boldsymbol{f}(u_S^i), \boldsymbol{g}(u_S^i) >_{\mathbb{R}^2} \\ &=\; \sum_{i=1}^{n} \boldsymbol{f}(u_S^i)^T \cdot \boldsymbol{g}(u_S^i), \end{aligned} \tag{12}$$

then the normal equations of the least-squares problem (9) are

$$\mathbf{N}\boldsymbol{\lambda} = \mathbf{d}, \tag{13}$$

where $N_{i,j} =< \mathbf{B}_i, \mathbf{B}_j >= \sum_{k=1}^{n} (\mathbf{B}_i u_S^k)^T \cdot \mathbf{B}_j u_S^k$ and $d_i =< \mathbf{B}_i, \overline{\psi} >= \sum_{k=1}^{n} (\mathbf{B}_i u_S^k)^T \cdot u_T^k$, with $i = 1, \dots, 4$ and $j = 1, \dots, 4$.

The matrix $\mathbf{N}$ is non-singular if and only if not all the points in U_S are aligned. Therefore, the system (13) has a unique solution, $\boldsymbol{\lambda}^0$. From the numerical solution, $\boldsymbol{\lambda}^0$, of the linear system (13) the linear transformation $\mathbf{A}^0 = \sum_{i=1}^{n} \boldsymbol{\lambda}_i^0 \mathbf{B}_i$ can be determined. Therefore, using $\mathbf{A}^0$ and equations (4) and (6), the following affine mapping can be established

$$\psi^0(u_S) := \mathbf{A}^0(u_S - u_S^c) + u_T^c \tag{14}$$

In conclusion, an affine mapping (14) between parametric spaces has been found that fits, in the least-squares sense, the loops of boundary data. This transformation can be used to map meshes from D_S to D_T. Finally, to obtain the mesh $\mathcal{M}_T$ it is only needed to map up the nodes onto the target surface T. To this end, an according to (2), we define

$$\widetilde{\psi}^0(p) := \varphi_T|_{D_T}\Big(\psi^0\big(\varphi_S|_{D_S}{}^{-1}(p)\big)\Big), \qquad p \in S. \quad (15)$$

Note that, since the values of $\varphi_S|_{D_S}{}^{-1}$ are known in all nodes of $\mathcal{M}_S$, the new mesh on the target surface can be defined as

$$\mathcal{M}_T := \widetilde{\psi}^0(\mathcal{M}_S).$$

2.3 Linking-sides mesh generation

Since linking-sides are always defined by four logical sides (each logical side can be composed by several edges), they can be meshed using any standard structured quadrilateral mesh algorithm, for instance, transfinite mapping (TFI) [2]. In order to apply the TFI method it is required that opposite logical sides will have the same number of nodes. It is important to have high quality structured meshes on the linking-sides. Note that these meshes determine the loops of nodes that are used later to generate the inner volume nodes in a layer by layer procedure. Thus, if these surface meshes contains folded or low quality elements, then tangled meshes, reverse oriented or low quality hexahedral elements are obtained.

A hard test for a sweep algorithm is to mesh an extrusion volume with an S-shaped changing sweep direction, see figure 4. It is well known that obtaining a good structured mesh over an S-shaped surface is non–trivial. If nodes are generated equidistant along the edges of the S-shaped surface, then some segments of the structured surface mesh cross over each other, see figure 4(a). Thus, folded quadrilateral elements are obtained in the middle part of the surface mesh. Therefore, tangled hexahedral elements are generated inside of the sweep volume.

To solve this drawback, a smoothing algorithm can be used. This smoother has to be able to untangle elements and improve the mesh quality by moving inner nodes and sliding the boundary nodes along the edges. Another alternative, such that no mesh smoothing is required, is to generate nodes along S-shaped edges in such a way that the distance between opposite nodes is minimized. Hence, we have implemented an edge mesher procedure that is able to "follow" nodes on the opposite edge of the S-shaped surface. Using this procedure, consecutive joining segments will not cross each other at the middle part of the surface, see figure 4(b). The details of this edge mesher are out of the scope of this work.

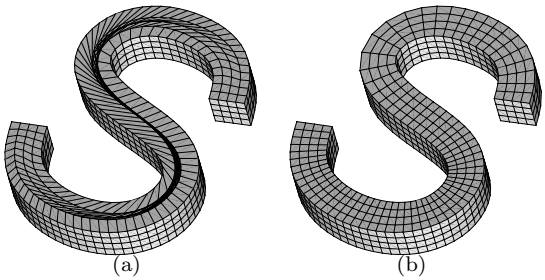

Figure 4: S-shaped sweep volume. **(a)** equidistant nodes on the edges and folded elements; **(b)** well positioned edge nodes for the linking-sides structured mesh generation.

2.4 Inner nodes and elements generation

Once all boundaries are meshed, inner nodes of the extrusion volume have to be generated. These nodes have to be placed, layer by layer, along the sweep direction. Each layer is delimited by a several loops of nodes that belongs to the structured meshes of the linking-sides (see figure 2). In fact, for every layer there is one outer loop, and one inner loop for each hole in the sweep volume. Note that the method developed in section 2.2 can not be used here because no surface parameterization, $\varphi_T|_{D_T}$, is available for these layers. To this end, the method developed in section 2.2 is extended to $\mathbb{R}^3$ and a weighted least-squares approximation is developed, similar to those proposed in [14, 15, 16].

Assume that $m-1$ inner levels of nodes have been generated on the linking-sides along the sweep direction (see figure 5). Therefore, $m-1$ layers of inner nodes have to be generated. Let $X_0 = \{x_0^i\}_{i=1,\dots,n} \subset \mathbb{R}^3$, $X_m = \{x_m^i\}_{i=1,\dots,n} \subset \mathbb{R}^3$ and $X_k = \{x_k^i\}_{i=1,\dots,n} \subset \mathbb{R}^3$ with $k = 1, \dots, m-1$ be the physical coordinates of the boundary nodes of: the source surface (level 0), the target surface (level m) and the k-th level, respectively. For a given level k, with $k = 1, \dots, m-1$, we look for a function ϕ such that

$$x_k^i = \phi(x_0^i), \qquad i = 1, \dots, n. \quad (16)$$

As in section 2.2, ϕ is approximated by an affine mapping, concretely

$$x_k = \phi(x_0) \approx \mathbf{A}(x_0 - x_0^c) + x_k^c, \quad (17)$$

where x_0 and x_k are points on the levels 0 and k respectively, $x_0^c = \frac{1}{n}\sum_{i=1}^n x_0^i$, $x_k^c = \frac{1}{n}\sum_{i=1}^n x_k^i$ and $\mathbf{A}$ is a linear transformation with its origin at x_0^c. For convenience, let

$$\overline{x}_0 = x_0 - x_0^c, \qquad \overline{x}_k = x_k - x_k^c. \quad (18)$$

Then, (17) can be expressed as

$$\overline{x}_k = \overline{\phi}(\overline{x}_0) \approx \mathbf{A}\overline{x}_0, \quad (19)$$

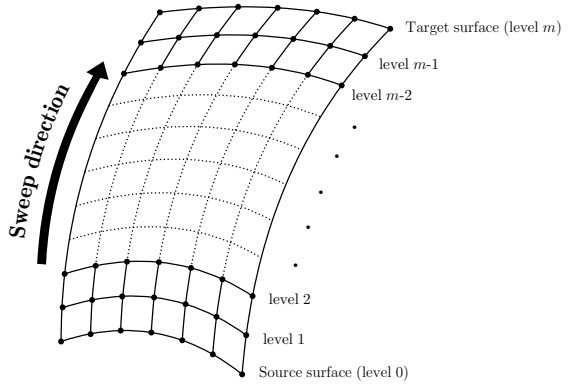

Figure 5: Discretization of a linking side using $m-1$ inner levels.

where $\overline{\phi}$ is the expression of $\overline{\phi}$ in the new coordinates (18). Similar to section 2.2, a least-squares fitting of the boundary data is performed in order to find a linear transformation, that minimizes

$$F(\mathbf{A}) = \sum_{i=1}^{n} \|\overline{x}_k^i - \mathbf{A}\overline{x}_0^i\|^2 \tag{20}$$

Since $\mathbf{A}$ is a linear transformation, it can be written as a linear combination of a base of the linear transformations from $\mathbb{R}^3$ to $\mathbb{R}^3$

$$\mathbf{A} = \sum_{i=1}^{9} \lambda_i \mathbf{B}_i, \tag{21}$$

where $\lambda_i \in \mathbb{R}$, with $i = 1, \ldots, 9$, and

$$\begin{aligned}
\mathbf{B}_1 &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \mathbf{B}_2 &= \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \\
\mathbf{B}_3 &= \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \mathbf{B}_4 &= \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \\
\mathbf{B}_5 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \mathbf{B}_6 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \\
\mathbf{B}_7 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, & \mathbf{B}_8 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \\
\mathbf{B}_9 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.
\end{aligned} \tag{22}$$

The normal equations of the least-squares problem (20) defined by the base (22) and the extension to $\mathbb{R}^3$ of the scalar product (12) are

$$\mathbf{N}\lambda = \mathbf{d}, \tag{23}$$

where $N_{i,j} =< \mathbf{B}_i, \mathbf{B}_j >$ and $d_i =< \mathbf{B}_i, \overline{\phi} >$, with $i = 1, \ldots, 9$ and $j = 1, \ldots, 9$.

Note that $\mathbf{N}$ is singular if and only if all points $\{\overline{x}_0^i\}_{i=1,\ldots,n}$ are coplanar with the origin. Unfortunately, if the initial loop of points $\{x_0^i\}_{i=1,\ldots,n}$ is planar, then $\{\overline{x}_0^i\}_{i=1,\ldots,n}$ are located on a plane that includes the origin. Therefore, $\mathbf{N}$ is singular in this case. To overcome this problem, the coordinate system proposed in [14] is used

$$\overline{x}_0 = x_0 - (2x_0^c - x_k^c), \qquad \overline{x}_k = x_k - x_0^c. \tag{24}$$

It is known that in these new coordinates the matrix $\mathbf{N}$ is not singular although the boundary of the source surface is planar (except for degenerated geometries). Using these new coordinates a least-squares approximation of the boundary data can be done such that equation (20) is also minimized. That is, the normal equations (23) can also be written in the new basis (24) and the same numerical method can be used to solve the 9×9 linear system.

Let $\boldsymbol{\lambda}^0$ be the computed solution of the linear system (23). Using (24), (21), and (19), a least squares approximation of the projection between the source surface and the k-th level can be written as

$$\phi_k^0(x_0) := \mathbf{A}^0(x_0 - 2x_0^c + x_k^c) + x_0^c \tag{25}$$

where $\mathbf{A}^0$ is the linear transformation corresponding to the solution $\boldsymbol{\lambda}^0$ by (21), and x_0 is any point of the source surface mesh.

Note that a similar process could be defined using the target surface as initial surface instead of the source surface. Hence, the nodes of the k-th level can be computed as

$$\phi_k^m(x_m) := \mathbf{A}^m(x_m - 2x_m^c + x_k^c) + x_m^c, \tag{26}$$

where x_m is any point of the target surface mesh.

Given two meshes, $\mathcal{M}_\mathcal{S}$ and $\mathcal{M}_\mathcal{T}$, over the source and target surfaces, respectively, equations (25) and (26) allows to obtain two different maps of these surfaces to the k-th level. Therefore, and similar to [16], the following weighted transformation is defined

$$\phi_k(p) := \left(1 - \frac{k}{m}\right)\phi_k^0(p) + \frac{k}{m}\,\phi_k^m\Big(\widetilde{\psi}^0(p)\Big), \tag{27}$$

where $p \in \mathcal{M}_S$, $k = 1, \ldots, m-1$, and $\widetilde{\psi}^0$ is the transformation defined in (15). For each level, we have to compute ϕ_k^0 and ϕ_k^m. In order to reduce the computational cost of the method, and do not recompute the set $\{\widetilde{\psi}^0(p) \in \mathbb{R}^3 \mid p \in \mathcal{M}_S\}$ for each level, the mapping relationship between nodes on $\mathcal{M}_\mathcal{S}$ and their images on $\mathcal{M}_\mathcal{T}$ can be stored once source surface mesh is projected on the target surface. In fact, it suffices to store the sequence of nodes in $\mathcal{M}_\mathcal{T}$ in the same order as the corresponding nodes in $\mathcal{M}_\mathcal{S}$.

Thus, given a mesh $\mathcal{M}_S$ over the source surface, the mesh $\mathcal{M}_k$ of the k-th level is computed as

$$\mathcal{M}_k := \phi_k(\mathcal{M}_S), \qquad k = 1, \ldots, m. \tag{28}$$

Finally, hexahedral elements are generated by joining the corresponding nodes between adjacent layers of quadrilateral meshes.

3. EXAMPLES

In order to asses the quality of the sweep algorithm described above, seven examples are presented. They illustrate the capabilities of the new algorithm to mesh an extrusion geometry defined by *1.*- any CAD application; *2.*- non linear sweeping trajectories; *3.*- non constant cross section along the sweep axis; *4.*- non parallel cap surfaces; and *5.*- cap surfaces with different shape and curvature. Moreover, these examples shows that the layers of inner nodes are distributed in such a way that a smooth transition between the curvatures of cap surfaces is obtained. Finally, they also illustrate that the developed algorithm, coupled with volume decomposition, can be successfully used to mesh a large class of three dimensional geometries.

In the first example a spring is meshed. The geometry is defined using a commercial CAD software. The user assigns the element size, and the application automatically determines the non linear sweeping direction. The final mesh is presented in figure 6(a). It is composed by 22740 hexahedral elements. As it can be seen, a non-structured quadrilateral mesh is generated over the cap surfaces. Note that, although the sweeping axis is non linear, high quality elements are generated without a posteriori mesh smoothing.

The second example presents the discretization of a bore bit with linear sweeping axis and rotated cap surfaces. The mesh, see figure 6(b), is composed by 21276 elements. Note that high quality elements are generated although the twisted extrusion path. Smoothing of the final hexahedral mesh was not required.

In the third example, the extrusion volume is defined by two cap surfaces with a different curvature. Figure 6(c) shows the inner elements of the 2000 hexahedra final mesh. Note that the weighted function (27) generates layers of elements with a smooth transition of the curvature from the source surface to the target surface. Notice that no smoothing was applied to obtain this mesh.

The fourth example shows the discretization of a extrusion volume defined by varying and rotating cross-sections along the sweep path. These cross-sections are elliptical-shaped with different size, and just on the middle of the extrusion path becomes circular. Moreover, the cap surfaces are rotated 90 degrees. The final mesh is composed by 2373 elements, see figure 7(a). It is important to point out that to obtain the final mesh no smoothing was applied on surface and inner nodes. In order to show the quality of elements inside the vol-

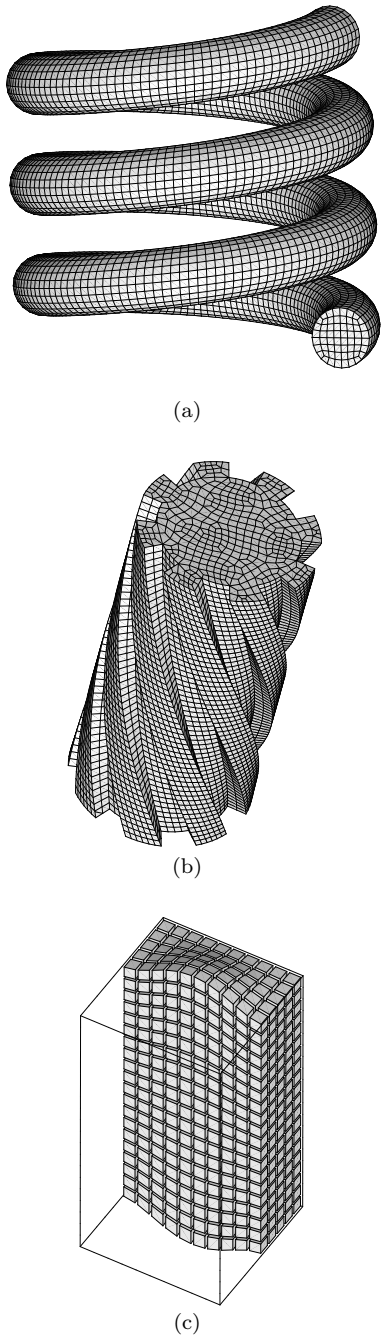

Figure 6: Examples of extrusion geometries meshed with the developed sweep algorithm. **(a)** spring; **(b)** bore bit; **(c)** Extrusion volume with non planar cap surfaces.

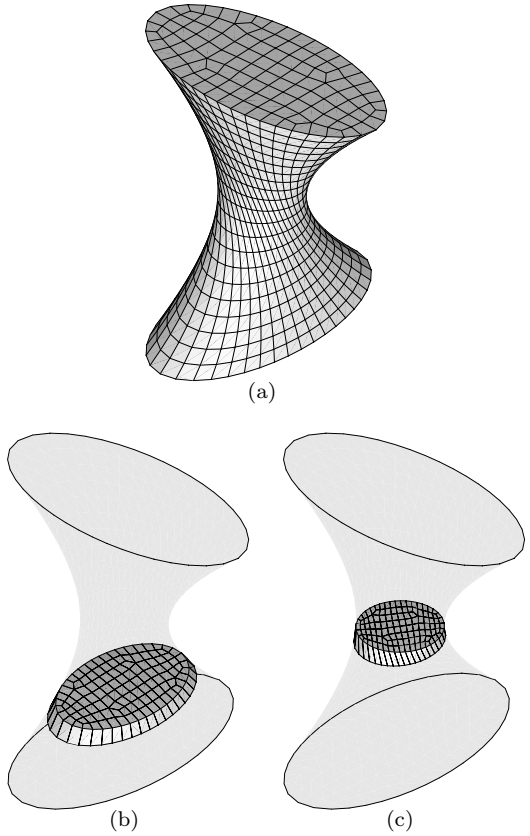

Figure 7: Volume with varying elliptical cross-sections along a twisted sweep path. **(a)** Layer of hexahedral elements at fourth-level; **(b)** middle layer of hexahedral elements.

ume, figure 7(b) shows the fourth layer of hexahedral elements and figure 7(c) presents the middle layer with one circular bounding loop.

The fifth example shows an application of the developed algorithm to an extrusion volume defined by two non-affine cap surfaces. In this case, a smoothing algorithm had been applied on the target surface mesh. Note that in this example source surface is convex and target surface has concavities. Therefore, folded quadrilateral elements appear near the concavity of the target surface. In order to unfold these quadrilateral elements it is mandatory to smooth the target surface mesh. For this example we have used the smoothing technique presented in [28]. In figure 8(a) the whole mesh, composed by 8000 hexahedral elements, is presented. Several inner layers are showed in figures 8(b), 8(c) and 8(d). Although the cap surfaces ar not affine, no additional 3D global smoothing algorithm has been required in this case.

The sixth example presents a sweep volume with non-

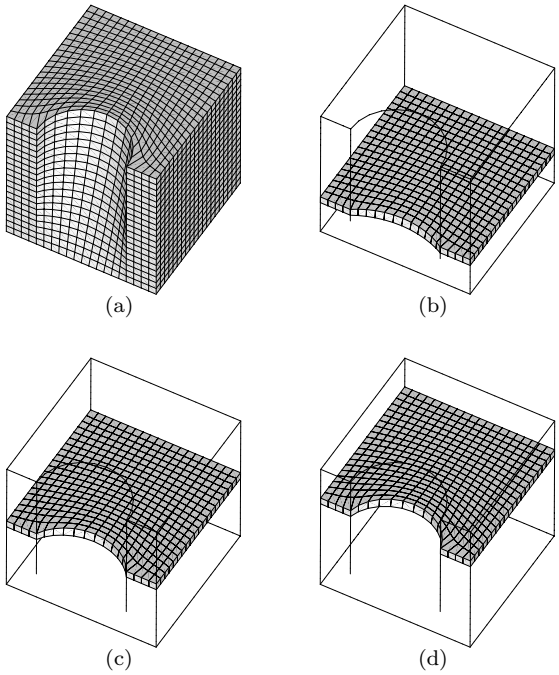

Figure 8: Sweep volume with convex source face and a target face with some concavities. **(a)** Whole mesh; **(b)** layer of hexahedral elements at level six; **(c)** layer of hexahedral elements at level eleven; **(d)** layer of hexahedral elements at level fifteen.

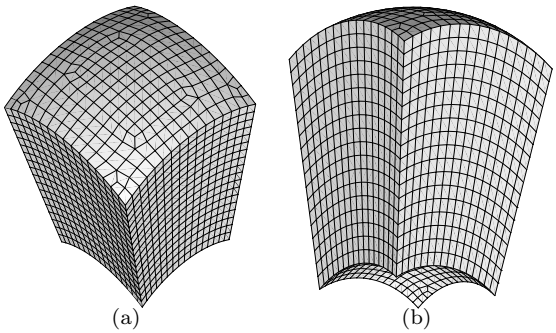

Figure 9: A sweeping volume with curved source and target surfaces. **(a)** Top view of the final mesh; **(b)** Lateral view the final mesh.

planar source and target surfaces. Moreover, the boundary loops of these surfaces, and the loops of boundary nodes for the inner layers, are non-planar too. The obtained high quality discretization is composed by 4320 elements, see figure 9. Notice that no mesh smoothing is applied to obtain the target surface mesh and the final hexahedral mesh.

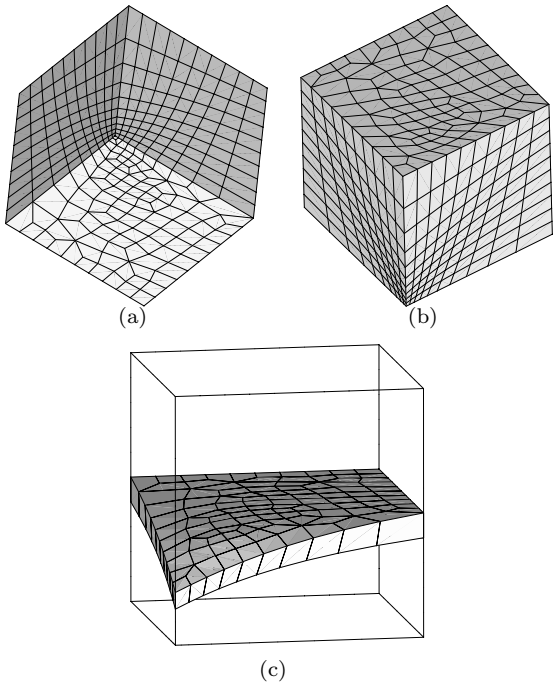

Figure 10: Cube with non uniform element size distribution. **(a)** Detail of the final mesh at the corner where high element concentration is prescribed; **(b)** View of target surface mesh; **(c)** Inner layer of hexahedral elements.

The seventh example shows the discretization of a cube with a non–constant element size distribution. A high element concentration is prescribed at one corner of the source surface. Hence, the boundary loops of nodes are not mutually affine. Figure 10 shows the final mesh composed by 1090 elements. Two smoothing steps were required: the first one to smooth the target surface mesh, and the second one to improve the overall quality of the hexahedral mesh.

The eighth example presents the application of the developed algorithm to an extrusion geometry composed by several sweep volumes. Figure 11(a) shows the discretization of a gear. The mesh is composed by 13712 elements. A conformal mesh is generated over the shared surfaces that compose the pieces of the volume.

The last example shows the generated mesh for a crank shaft model, see figure 11(b). It is composed by 15800 elements. As in the previous example, a conformal mesh is obtained for this multi-block geometry.

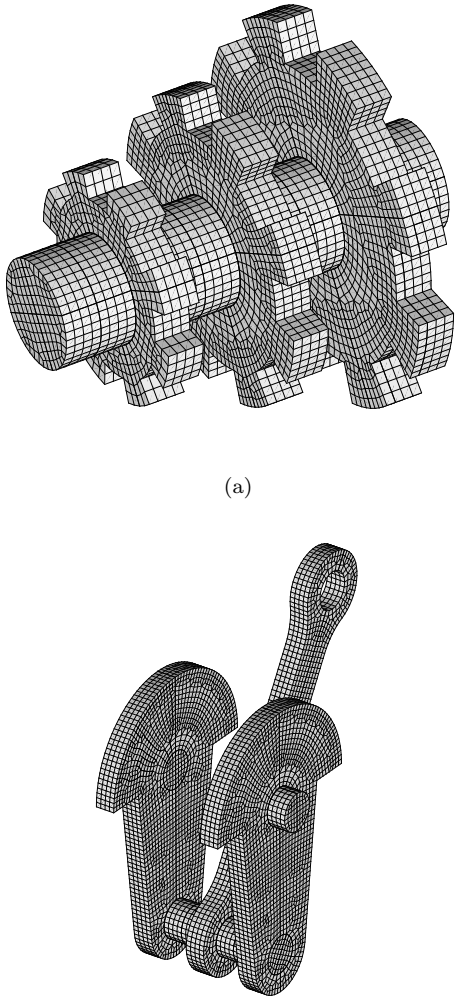

Figure 11: Examples of extrusion geometries meshed with the developed sweep algorithm. **(a)** gear; **(b)** crank shaft.

4. CONCLUSIONS

In this paper a new algorithm to project meshes between two topologically equivalent surfaces has been presented. It has been successfully implemented in a sweep tool to mesh extrusion geometries. This projection is determined by means of a least-squares approximation of a transformation defined between the loops of boundary nodes of the cap surfaces in the parametric spaces. Once the new mesh is obtained in the parametric space, it is mapped up according the target surface parameterization. This projection algorithm has been extended to three dimensional spaces. Then, it has been used to generate the inner nodes

for a sweep tool. In this case, the inner nodes are located using a weighted least-squares approximation of the transformation between the boundary nodes of the cap surfaces and the boundary nodes of the layer. Finally, several examples have been presented to show the new algorithm capabilities.

References

[1] Tautges T.J. "The generation of hexahedral meshes for assembly geometry: survey and progress." *International Journal for Numerical Methods in Engineering*, vol. 50, no. 12, 2617–2642, 2001

[2] Thompson J.F., Warsi Z.U.A., Mastin C.W. *Numerical grid generation: foundations and applications.* Elsevier North-Holland, 1985

[3] Schneiders R. "A grid-based algorithm for the generation of hexahedral element meshes." *Engineering with Computers*, vol. 12, no. 3-4, 168–177, 1996

[4] Taghavi R. "Automatic, parallel and fault tolerant mesh generation from CAD." *Engineering with Computers*, vol. 12, no. 3-4, 178–185, 1996

[5] Dhondt G. "A new automatic hexahedral mesher based on cutting." *International Journal for Numerical Methods in Engineering*, vol. 50, no. 9, 2109–2126, 2001

[6] Price M.A., Armstrong C.G., Sabin M.A. "Hexahedral Mesh Generation by Medial Surface Subdivision: Part I. Solids with Convex Edges." *International Journal for Numerical Methods in Engineering*, vol. 38, no. 19, 3335–3359, 1995

[7] Price M.A., Armstrong C.G. "Hexahedral mesh generation by medial surface subdivision: Part II. Solids with flat and concave edges." *International Journal for Numerical Methods in Engineering*, vol. 40, no. 1, 111–136, 1997

[8] Taghavi R. "Automatic block decomposition of parametrically changing volumes." *Oil & Gas Science and Technology-Revue De L Institut Francais Du Petrole*, vol. 54, no. 2, 193–196, 1999

[9] Blacker T.D., Meyers R.J. "Seams and Wedges in Plastering - a 3-D Hexahedral Mesh Generation Algorithm." *Engineering with Computers*, vol. 9, no. 2, 83–93, 1993

[10] Tautges T.J., Blacker T., Mitchell S.A. "The whisker weaving algorithm: A connectivity-based method for constructing all-hexahedral finite element meshes." *International Journal for Numerical Methods in Engineering*, vol. 39, no. 19, 3327–3349, 1996

[11] Calvo N.A., Idelsohn S.R. "All-hexahedral element meshing: Generation of the dual mesh by recurrent subdivision." *Computer Methods in Applied Mechanics and Engineering*, vol. 182, no. 3-4, 371–378, 2000

[12] Cook W.A., Oakes W.R. "Mapping methods for generating three-dimensional meshes." *Computers In Mechanical Engineering*, pp. 67–72, 1982

[13] White D. *Automatic, quadrilateral and hexahedral meshing of pseudo-cartesian geometries using virtual subdivision.* Master in science, Brigham Young University, 1996

[14] Knupp P.M. "Next-Generation Sweep Tool: A Method For Generating All-Hex Meshes On Two-And-One-Half Dimensional Geomtries." *7th International Meshing Roundtable*, pp. 505–513. Sandia National Lab, 1998

[15] Knupp P.M. "Applications of mesh smoothing: Copy, morph, and sweep on unstructured quadrilateral meshes." *International Journal for Numerical Methods in Engineering*, vol. 45, no. 1, 37–45, 1999

[16] Blacker T. "The Cooper Tool." *5th International Meshing Roundtable*, pp. 13–30. Sandia National Laboratories, 1996

[17] Lai M., Benzley S., White D. "Automated hexahedral mesh generation by generalized multiple source to multiple target sweeping." *International Journal for Numerical Methods in Engineering*, vol. 49, no. 1-2, 261–275, 2000

[18] Staten M.L., Canann S.A., Owen S.J. "BMSweep: Locating interior nodes during sweeping." *Engineering with Computers*, vol. 15, no. 3, 212–218, 1999

[19] Blacker T.D., Stephenson M.B. "Paving - a New Approach to Automated Quadrilateral Mesh Generation." *International Journal for Numerical Methods in Engineering*, vol. 32, no. 4, 811–847, 1991

[20] Cass R.J., Benzley S.E., Meyers R.J., Blacker T.D. "Generalized 3-D paving: An automated quadrilateral surface mesh generation algorithm." *International Journal for Numerical Methods in Engineering*, vol. 39, no. 9, 1475–1489, 1996

[21] Sarrate J., Huerta A. "Efficient unstructured quadrilateral mesh generation." *International Journal for Numerical Methods in Engineering*, vol. 49, no. 10, 1327–1350, 2000

[22] Sarrate J., Huerta A. "Automatic mesh generation of nonstructured quaadrilateral meshes over curved surfaces in $\mathbb{R}^3$." *European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS*. Barcelona, Spain, 2000

[23] Haber R., Shephard M.S., Abel J.F., Gallagher R.H., Greenberg D.P. "A General Two-Dimensional, Graphical Finite-Element Preprocessor Utilizing Discrete Transfinite Mappings." *International Journal for Numerical Methods in Engineering*, vol. 17, no. 7, 1015–1044, 1981

[24] Goodrich D. *Generation of all-quadrilateral surface meshers by mesh morphing.* Master in science, Brigham Young University, 1997

[25] White D.R., Tautges T.J. "Automatic scheme selection for toolkit hex meshing." *International Journal for Numerical Methods in Engineering*, vol. 49, no. 1-2, 127–144, 2000

[26] Miyoshi K., Blacker T. "Hexahedral Mesh Generation Using Multi-Axis Cooper Algorithm." *9th International Meshing Roundtable*, pp. 89–97. Sandia National Laboratories, Albuquerque, New Mexico, 2000

[27] Hatcher A. *Algebraic topology.* Cambridge University Press, New York, 2002

[28] Sarrate J., Huerta A. "A new approach to minimize the distortion of quadrilateral and hexahedral meshes." *4th European congress on computational methods in applied sciences and engineering.* Jyväskylä, Finland, 2004